Programando Redes de Sensores Inalámbricos en la Forma Web 2.0

Eduardo Omar Sosa; Sergio Fabián Vier
Secretaría de Investigación y Posgrado; Facultad de Ciencias Exactas, Químicas y Naturales; Universidad Nacional de Misiones.
Félix de Azara 1552 - C.P. N3300LQH - Posadas - Misiones
{eososa , svier}@fceqyn.unam.edu.ar

Resumen

La programación de redes inalámbricas de sensores (WSN) es un proceso propenso a errores. Cada sensor de la red contiene una gran variedad de componentes que han de ser programados con diferentes métodos y/o lenguajes de programación. Las aplicaciones Web 2.0 se enfrentan a problemas similares. Google Web Toolkit (GWT) es una herramienta para Aplicaciones Web 2.0, que permite desarrollar la aplicación completa en sólo un lenguaje de programación, y sin necesidad de conocimiento de las diferentes técnicas existentes.

En este trabajo se pretende utilizar los conceptos de GWT, para intentar una posible adaptación/ modificación, de tal manera que pueda realizarse la programación de las WSN de una nueva forma; en la forma Web 2.0.

Palabras clave: Redes de Sensores, Redes ad hoc, programación, web 2.0, WSN, GWT.

Contexto

Este Proyecto ha sido presentado en la Secretaría de Investigación y Posgrado de la Facultad de Ciencias Exactas, Químicas y Naturales (UNaM), habiendo recibido el código Nº 457/10.

Se articula en cuanto a contenidos y prospectiva con el proyecto “Hacia una Red Global de Sensores Interconectados, un ensayo experimental Argentino-Alemán”, presentado en el marco del Programa de Cooperación Científico-Tecnológico entre el Ministerio de Ciencia, Tecnología e Innovación Productiva de la República Argentina (MINCYT) y el Bundesministerium für Bildung und Forschung (BMBF) de Alemania que ha sido aprobado bajo el código AL0807.

Introducción

Las WSN son el resultado de los múltiples avances tecnológicos logrados en electrónica, nanotecnología, comunicaciones inalámbricas, potencia de cálculo, desarrollo de redes y robótica. Componen sistemas distribuidos, normalmente compuestos de dispositivos integrados, incluyendo como mínimo CPU, radio, y sensores/actuadores varios.

Las redes de sensores inalámbricos (WSN) son redes heterogéneas conformadas por sensores, gateways y backends de recursos

físicos muy limitados. Los sensores pueden medir parámetros como temperatura, movimiento, iluminación, humedad, etc.; los gateways establecen el nexo con las redes tradicionales y conocidas. Los backends son responsables por el proceso y visualización de los datos capturados. El desarrollo de aplicaciones para WSN es tarea compleja, debiendo plasmarse en aplicaciones distribuidas e integradas, existiendo complicaciones extras como la heterogeneidad, influencias ambientales y la escala.

Si bien diversos trabajos presentaron middleware para WSN, no se ha logrado con ello aceptación en la industria, debido principalmente a las diferentes metodologías de programación.

Las WSN consisten en equipos de muy bajos consumo, costos y factores de forma. La realidad es bastante diferente a entornos donde las aplicaciones están respaldadas con equipamiento más poderoso y alimentados por redes de energía eléctrica redundantes.

El trabajo pretende definir una forma en la que programadores pueden utilizar tecnologías estándar compatibles existentes, para describir los procesos y servicios ofrecidos por WSN, sin codificación extra. Las ventajas del método son dos: inmediata adaptación de las WSN a empresa, e integración de los servicios de las WSN a aplicaciones de nivel superior mediante modificaciones simples.

La realidad indica que los principales inconvenientes encontrados en la programación de WSN tienen que ver con las diferentes metodologías de programación, como también con la heterogeneidad existente tanto en hardware como en sistemas operativos (1). Mientras que en las empresas y oficinas las aplicaciones se encuentran respaldadas con equipamiento de gran potencia de cálculo y son alimentadas por redes de energía redundantes; las WSN son optimizadas para escenarios de mínimos consumos energéticos, bajos costos y reducidos factores de forma. En entornos corporativos ó gubernamentales es importante adaptar los procesos de negocio a la infraestructura subyacente de software, para así poder reaccionar rápidamente frente a potenciales cambios y exigencias de los mercados.

Desde principios de la década del 90, se ha buscado lograr éste objetivo con el modelado, análisis y adaptación de los procesos de negocio. Así han aparecido las arquitecturas orientadas al servicio, las que se basan en Internet (SOA (2)). Paralelamente, las WSN han logrado un grado tal de desarrollo, que se consideran como parte integral de la Internet del futuro, logrando extender el dominio de ésta red al mundo real. Las dos tendencias, en conjunto, forman la base de un nuevo tipo de aplicaciones donde los dispositivos interactúan en procesos de una manera imposible de imaginar hace unos pocos años. Lo anterior se cumple para simple nodos sensores hasta para servidores de aplicación de gran escala,

Con ello, y si la tendencia no declinara, los datos capturados por las WSN influenciarían el flujo de información de los procesos actuales en tiempo real, e incluso podrían disparar nuevos procesos. Para lograr este nivel de interacción, las WSN deben relacionarse, indefectiblemente, con las actuales tecnologías SOA existentes, tales como XML (3), Servicios Web (4), y el lenguaje de ejecución de procesos de negocio (BPEL (5)), para nombrar solamente algunos. Sin embargo, debido a la elevada demanda de recursos de estas aplicaciones, son difícilmente aplicables en los entornos restringidos de las WSN.

Algunas características comunes de las aplicaciones en WSN son definidas por la taxonomía propia de estas redes, como se indica en la Fig. 1.

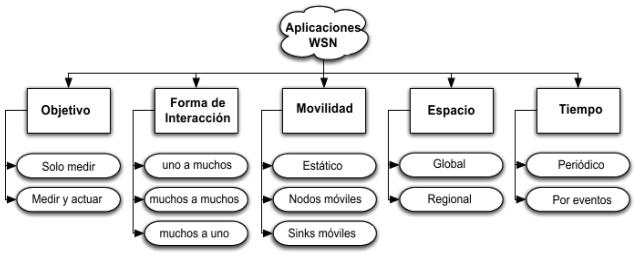

Fig. 1 Taxonomía de Aplicaciones WSN

No es una buena idea añadir soporte para cada una de las plataformas en las herramientas de desarrollo. Esto se racionaliza con la adición de una capa de integración entre el hardware y la aplicación, conocida como middleware. El middleware ha sido tema de investigación de sistemas distribuidos por muchos años. Crear un middleware para una WSN en un desafío, considerando las restricciones presentadas por ésta tecnología, métodos bien conocidos como CORBA ó Enterprise Java Beans deben ser descartados por excesivo requerimientos de potencia de cálculo y memoria. Además, por la inestabilidad de las comunicaciones en el ambiente de las WSN, los métodos tradicionales cliente-servidor no son recomendados (6), (7), (8).

El hardware es importante en las WSN, pero la tecnología puede ser plenamente aprovechada solamente si las plataformas de software se encuentran a disposición de los desarrolladores. Muy pocas aplicaciones para WSN en el mundo real son de alto nivel (9), (10). En la mayoría de los casos el desarrollo de las aplicaciones se encuentra próximo al sistema operativo, por lo que se debe lidiar con situaciones que atañen al bajo nivel, tales como protocolos distribuidos. Estas capacidades son poco frecuentes en los desarrolladores actuales.

Una red de sensores típica está formada generalmente por cientos y en algunos casos miles de nodos. En situaciones reales es muy dificultoso establecer un backbone cableado, el que se utilice para la programación de todos y cada uno de los nodos, convirtiéndose esta en una tarea de alta dificultad.

Si bien el enfoque de este trabajo permite la integración de las aplicaciones de WSN a aquellas típicas del dominio IT, los programadores deben ser expertos en programación WSN para aplicar los Servicios Web en entornos restrictivos en cuanto a recursos.

El desarrollo de aplicaciones para WSN es comparable con la situación similar observada en la programación de de aplicaciones Web 2.0. Google (11) publica: “Actualmente, la creación de aplicaciones web resulta un proceso pesado y propenso a errores. Los desarrolladores pueden pasar el 90% de su tiempo estudiando las peculiaridades de los navegadores. Por otra parte, la creación, la reutilización y el mantenimiento de una gran cantidad de componentes AJAX y bases de código Java Script pueden ser tareas complejas y delicadas”.

Para facilitar esta tarea, Google ofrece el llamado "Google Web Toolkit (GWT)" ayudando a los programadores en la implementación de aplicaciones Web. Con GWT, los desarrolladores pueden utilizar el lenguaje de programación Java para ejecutar tanto el servidor como navegador en una sola aplicación. Por el lado del servidor, se puede aprovechar toda la funcionalidad proporcionada por Java, incluyendo las numerosas bibliotecas, frameworks y herramientas. Por el lado del navegador, existen algunas restricciones, siendo

la principal que sólo un subconjunto de la API de Java puede ser utilizado.

El servidor y el navegador no se comunican a través de llamadas a funciones, sino mediante el envío de las instancias de los objetos entre sí. Estos objetos deben ser serializados siguiendo la especificación de Java (12). GWT, después de compilar el código Java a Java Script, el ejecutable es capaz de funcionar en todos los navegadores principales (IE, Firefox, Opera, etc.). Esta es la razón para el limitado conjunto de APIs disponibles en el lado del cliente, dado que a los navegadores se le ofrecen solamente un conjunto limitado de posibilidades. Las clases y métodos compatibles, se encuentran en el GWT JRE Emulation Reference (13).

La belleza de este enfoque se encuentra en que los programadores se benefician de las herramientas de Java, como los IDEs, depuradores, escritura de código, etc.; logrando un ejecutable en un solo lenguaje. El del programa resultante está, normalmente, aún más optimizado que el código original.

Líneas de Investigación y Desarrollo

Las líneas de investigación convergen en estudios de los medios existentes, y sus bases conceptuales para relacionarlos con métodos y objetivos de campo de las WSN. Fundamentalmente en el contexto donde se han desarrollado herramientas exitosas de programación en la forma web 2.0.

De esta manera se pretende analizar y evaluar la posible adaptación tecnológica de: plataformas, compiladores, API y herramientas de producción de servicios Web.

Resultados y Objetivos

El objetivo de éste trabajo es proporcionar un conjunto de herramientas para el desarrollo de aplicaciones de WSN, en forma similar a lo que GWT ofrece para aplicaciones Web. Se pretende implementar el componenente que genere la aplicación WSN (WSA) y la aplicación que se ejecuta en Internet en Java (IAP) de la manera realizada actualmente para servidor y navegador en la Web.

Se pretende avanzar en el manejo de las siguientes tecnologías y conceptos: estudio de la Plataforma de sensores iSense (14), compilador GWT, Java API de Servicio Web y concepto de anotación, microFibre y Fabric (15) y LTP (16) proveyendo servicios Web para las WSN.

Se considera establecer una serie de interfaces de Java que abstraigan las funcionalidades de un nodo sensor. Esta API deberá proveer funcionalidad de: a) Interfaz de radio para enviar, recibir, registrar las devoluciones de llamadas, etc. b) Ruteo de la API c) Temporizadores d) Interfaz serial e) Depuración f) Sensores. Se debe determinar si el compilador de GWT es suficientemente genérico para que pueda ser modificado y así poder utilizar la nueva definición de las interfaces de sensores en Java y generar el código fuente en lenguaje destino.

Una de las lecciones aprendidas durante el correr del proyecto, es que tal vez el enfoque adoptado por GWT es la más adecuada para nuestro objetivo común, pero no GWT en sí mismo, debido a su extrema complejidad. Se considera analizar diversos módulos, como Eclipse JDT (17), que permitan convertir código fuente Java en un árbol de sintaxis abstracto (AST) y de esta forma poder construir el código requerido para poder operar con las WSN. Asimismo se analiza la reformulación de Fabric de manera que converjan en varios

proyectos distintos independientes uno del otro, cada uno de ellos con una finalidad definida.

Formación de Recursos Humanos

El grupo de trabajo se integra con un investigador formado, por tres investigadores en formación y un administrativo. En el marco de este proyecto se están desarrollando: una tesis doctoral y dos tesis de grado de licenciatura.

Referencias

1. *Programming Wireless Sensor Networks: Fundamental Concepts and State of the Art.* **Mottola, Luca and Pietro, Pico.** 4, s.l. : ACM, December 2010, ACM Computing Surveys, Vol. 43.

2. **Erl, Thomas.** *Service-Oriented Architecture (SOA): Concepts, Technology, and Design* Prentice Hall PTR, August 2005. Vol. 1. ISBN-13: 978-0131858589.

3. World Wide Web Consortium (W3C). *Recommendation: Extensible Markup Language (XML)* http://www.w3.org/TR/xml/.

4. **W3C Working Group.** Web Services Architecture Requirements. http://www.w3.org/TR/2004/NOTE-wsa-reqs-20040211/.

5. **OASIS WS-BPEL Technical Committee.** Web Services Business Process Execution Language Version 2.0. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html.

6. *Middleware challenges and approaches for wireless sensor networks.* **Mohamed, S. Hadim and N.** 3, s.l. : IEEE Distributed Systems Online, 2006, Vol. 7.

7. *A survey of middleware for sensor networks: State-of-the-art and future directions.* **Robinson, K. Henricksen and R.** 2006. Proc. of MidSens '06.

8. *Deployment of sensor networks: Problems and passive inspection.* **Romer, M. Ringwald and K.** Madrid . Proc. of the 5th Workshop on Intelligent Solutions in Embedded Systems.

9. *Monitoring heritage buildings with wireless sensor networks: The Torre Aquila deployment.* **Ceriotti, M., et al.** s.l. : IEEE, 2009. Proc. of the 8th ACM/IEEE Int. Conf. on Information Processing in Sensor Networks.

10. *Vineyard computing: Sensor networks in agricultural production.* **Buonadonna, P., et al.** IEEE Pervasive Computing 3, 1, 2004,.

11. **Google Inc.** Google Web Toolkit. [Online] 2011. http://code.google.com/intl/es-AR/webtoolkit/.

12. **Oracle, Sun Developer Network.** Discover the secrets of the Java Serialization API. http://java.sun.com/developer/technicalArticles/Programming/serialization/.

13. **Google Inc.** JRE Emulation Reference. [Online] http://code.google.com/intl/es-AR/webtoolkit/doc/1.6/RefJreEmulation.html.

14. **Coalesenses GmbH.** http://www.coalesenses.com.

15. **Pfisterer, D.** Fabric, [Online] 2010. https://github.com/pfisterer/fabric.

16. *LTP: An Efficient Web Service Transport Protocol for Resource Constrained Devices.* **N. Glombitza, D. Pfisterer, and S. Fischer.** 2010. Proc. SECON, 2010. pp. 199-207.

17. Eclipse JDT - Abstract Syntax Tree (AST) and the Java Model - Tutorial. [Online] http://www.vogella.de/articles/EclipseJDT/article.html.